Link Prediction in Networks

Vrushank Ahire
2022CSB1002

April 25, 2024

Abstract

This report presents a method for analyzing and predicting missing connections in a network dataset. The proposed approach utilizes Python libraries such as pandas, numpy, networkx, and matplotlib to process the data, construct a directed graph, and visualize the results. The primary objective is to identify missing connections within the network and predict their existence based on existing connections.

1 Introduction

Network analysis plays a crucial role in various fields, including social networks, computer networks, and biological networks. Understanding the structure and dynamics of these networks is essential for gaining insights and making informed decisions. This report focuses on analyzing a given network dataset and predicting missing connections within the network.

2 Methodology

2.1 Data Preprocessing

The first step involves reading the network data from an Excel file using the pandas library. The 'Roll Number' column is used to count the total number of unique IDs, which represent the nodes in the network. A directed graph is then created using the networkx library, with nodes ranging from 0 to (total_ids - 1).

2.2 Graph Construction

The node IDs are mapped to their corresponding 'Roll Number' values, and edges are created based on the connections specified in the dataset. For each node, the row index in the DataFrame is identified, and the columns (excluding 'Name' and 'Roll Number') are iterated over to find the connected roll numbers. The corresponding nodes are then connected by adding edges to the graph.

2.3 Missing Connection Identification

To identify missing connections, the adjacency matrix of the graph is computed using the nx.to_numpy_array() function from networkx. The missing connections are marked with -1 in both directions (for a pair of nodes) if there is no edge between them and they are not the same node.

2.4 Connection Prediction

The predict_link() function is implemented to predict missing connections using a least squares approach. For each pair of nodes with a missing connection (-1 in the adjacency matrix), the function extracts submatrices from the adjacency matrix and solves for x in the equation $Ax = B$ using the numpy.linalg.lstsq() function. The result is then used to determine whether a connection should be predicted (1) or not (0).

2.4.1 Algorithm

The predict_link() function follows the algorithm outlined below:

Algorithm 1 predict_link(matrix, row, col)

1. Extract submatrices A, B, C, and M from *matrix*
2. Solve $Ax = B$ using least squares: $x = \text{numpy.linalg.lstsq}(C, B, rcond = None)[0]$
3. Reshape x to a column vector
4. Compute $result = M^T \cdot x$
5. If $result > 0$, **return** 1; else, **return** 0

2.4.2 Mathematical Analysis

The predict_link() function employs a least squares approach to predict missing connections. Let A be the adjacency matrix up to row $r-1$ and column $c-1$, B be the row vector up to column $c-1$ for row r, and M be the column vector up to row $r-1$ for column c. The goal is to find a vector x that minimizes the residual $\|Ax - B\|_2$.

The normal equations for the least squares problem are given by:

$$A^T Ax = A^T B \tag{1}$$

Rearranging the terms, we get:

$$x = (A^T A)^{-1} A^T B \tag{2}$$

However, computing the inverse $(A^T A)^{-1}$ can be numerically unstable. Instead, we use the numpy.linalg.lstsq() function, which computes the least squares solution using QR factorization or SVD, depending on the conditioning of the problem.

Once the solution x is obtained, we calculate the result as $M^T x$. If the result is positive, a connection is predicted (1); otherwise, no connection is predicted (0).

3 Results and Discussion

The proposed method successfully identifies missing connections in the network dataset and predicts their existence based on the existing connections. The visualizations provide a clear representation of the original and resulting matrices, allowing for easy comparison and analysis.

4 Conclusion

This report presents a comprehensive approach to network analysis and prediction of missing connections. The proposed method leverages various Python libraries to preprocess the data, construct a directed graph, identify missing connections, predict potential connections, and visualize the results.

5 Visualization

The original and resulting adjacency matrices are visualized using matplotlib. The matrices are plotted as images, with color maps representing the connection values. The identified connections are also printed in the format "node_a node_b".

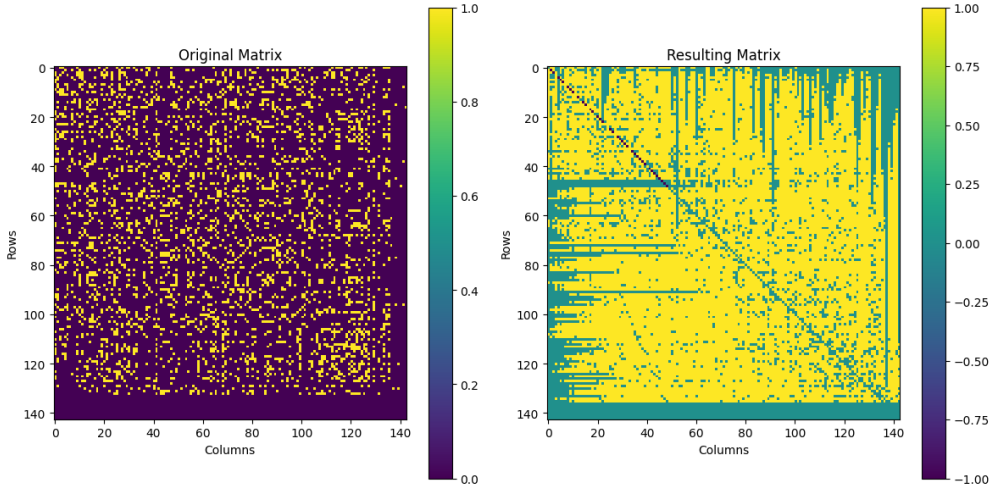

Figure 1: Matrix Comparison